Multi-resource Fair Allocation with Bounded Number of Tasks in Cloud Computing Systems

Weidong Li, Xi Liu, Xiaolu Zhang, and Xuejie Zhang

Abstract—Dominant resource fairness (DRF) is a popular mechanism for multi-resource allocation in cloud computing systems. In this paper, we consider a problem of multi-resource fair allocation with bounded number of tasks. Firstly, we propose the lexicographically max-min normalized share (LMMNS) fair allocation mechanism, which is a natural generalization of DRF, and design a non-trivial optimal algorithm to find a LMMNS fair allocation, whose running time is linear in the number of users. Secondly, we prove that LMMNS satisfies envy-freeness (EF) and group strategy-proofness (GSP), and analysis the approximation ratios of LMMNS, by exploiting the properties of the optimal solution. Thirdly, we propose a modified version of LMMNS, which is the second mechanism satisfying sharing incentive, EF, and GSP. Finally, we have implemented LMMNS, and show that it has a good average-case performance, especially when the number of resources is 2.

Index Terms—Lexicographically max-min normalized share, dominant resource fairness, multi-resource fair allocation, approximation ratio.

1 Introduction

Multi-resource fair (or efficient) allocation is a fundamental problem in any shared computer system including cloud computing systems. As pointed by Ghodsi et al. [1], the traditional slot-based scheduler for state-of-the-art cloud computing frameworks (for example, Hapdoop) can lead to poor performance, unfairly punishing certain workloads. Ghodsi et al. [1] are the first to suggest a compelling alternative known as the *dominant resource fairness* (DRF) mechanism, which is to maximize the minimum dominant share of users, where the dominant share is the maximum share of any resource allocated to that user. DRF is generally applicable to multi-resource environments where users have heterogeneous demands, and is now implemented in the Hadoop Next Generation Fair Scheduler [1].

In recent years, DRF has attracted much attention and been generalized to many dimensions. Joe-Wong et al. [2] designed a unifying multi-resource allocation framework that captures the trade-offs between fairness and efficiency, which generalizes the DRF measure. Parkes et al. [3] extended DRF in several ways, including the presence of zero demands and the case of indivisible tasks. Bhattacharya et al. [4] adapted the definition of DRF to suport hierarchies. Zeldes and Feitelson [5] proposed an online algorithm based on bottlenecks and global priorities. Kash, Procaccia and Shah [6] deveoped a dynamic model of fair division and proposed some dynamic resource allocation mechanisms based on DRF. Wang et al. [7] generalized the DRF measure into the cloud computing systems with heterogeneous servers. Psomans and Schwartz [8], Friedman, Ghodsi, and Psomas [9] studied the multi-resource allocation of discrete tasks on multiple machines. We refer to [10] to find other related works.

Notably, Dolev et al. [11] suggested another notion of fairness, called Bottleneck-Based Fairness (BBF), which guarantees that each user either receives all he wishes for, or else gets at least his entitlement on some bottleneck resource. Gutman and Nisan [12] situated DRF and BBF in a common economics framework, obtaining a general economic perspective. They [12] also presented a polynomial-time algorithm that computes a fair allocation for a general class of fairness notions including DRF, and by showing that a competitive market equilibrium achieves BBF, obtained a polynomial-time algorithm that computes a BBF solution, based on the previous algorithm that finds a competitive equilibrium in a Fisher Market for Leontief utilities. Very recently, Bonald and Roberts [13] argued that proportional fairness is preferable to DRF, especially assuming that the population of jobs in progress is a stochastic process. They also [14] introduced the bottleneck max fairness for the case of router resources allcoation, which is a variant of BBF. Zahedi and Lee [15] applied the concept of Competitive Equilibrium from Equal Outcomes (CEEI) in the case of Cobb-Douglas utilities to achieve properties similar to DRF.

As mentioned by Wang, Liang and Li [7], the users have a finite number of tasks in a real-world cloud computing system. They [7] studied the multi-resource allocation problem in heterogeneous cloud computing systems with bounded number of tasks, and proposed a generalized version of the well-known water-filling algorithm. However, the running time of the water-filling is pseudo-polynomial. It is desired to design an efficient algorithm. In fact, this is not the only paper studied multi-resource allocation problem with bounded number of tasks. In BBF [11], a user u_i receiving all he wishes for is equivalent to that all the tasks (finite) of u_i are processed. Very recently, Tang et al. [16] studied the multi-resource fair allocation in pay-as-you-go

• *W. Li, X. Liu, X. Zhang, and X. Zhang are with Yunnan University, Kunming, P.R. China, 650091.*
Correspondence: weidong@ynu.edu.cn (W. Li), xjzhang@ynu.edu.cn (X. Zhang).

Manuscript received XX XX, 2015; revised XX XX, 2015.

1. http://hadoop.apache.org/docs/r2.3.0/hadoop-yarn/hadoop-yarn-site/FairScheduler.html

cloud computing, where the number of tasks is bounded for each user at a given time, too.

In this paper, we consider the multi-resource allocation problem in a single server with bounded number of tasks, which is a special case of the model studied in [7]. The rest of the paper is organized as follows. Section 2 gives the motivations for our work. Section 3 introduces the lexicographically max-min normalized share (LMMNS) fair allocation mechanism, and presents a non-trivial polynomial-time optimal algorithm. Section 4 lists the allocation properties LMMNS satisfies, while Section 5 analyzes the efficiency of LMMNS. Section 6 presents the alternative mechanisms for multi-resource fair allocation. Section 7 provides a modified version of LMMNS, which satisfies all the desired properties. Section 8 provides experimental results based on random instances. Finally, Section 9 concludes the paper and gives the future work.

2 Motivation

In a cloud computing system, assume that we are given a server with m resources and n users. As in [1], [3], assume that each user u_i has a publicly known weight w_{ij} for resource j, which represents the amount of resource j contributed by user u_i to the resource pool. Without loss of generality, assume that $\sum_{i=1}^{n} w_{ij} = 1$, for $j = 1, \ldots, m$. Each user u_i requires r_{ij}-fraction of resource type j per task.

Let x_i be the number of tasks processed on the server for user u_i. The resource requirement constraints are

$$\sum_{i=1}^{n} r_{ij} x_i \le 1, \quad j = 1, \ldots, m. \tag{1}$$

The *weighted share* of resource j per task for user u_i is

$$ws_{ij} = \frac{r_{ij}}{w_{ij}}, \text{ for } j = 1, \ldots, m,$$

and the *dominant share* for user u_i is defined as

$$DS_i = x_i \max_j ws_{ij}.$$

The *dominant resource fairness* (DRF) mechanism [1] seeks to maximize the number of allocated tasks x_i, under the constraint that the dominant shares of the users are equalized, i.e.,

$$DS_1 = \cdots = DS_n. \tag{2}$$

The DRF allocation is equivalent to the solution for the following linear program:

$$\max \quad (x_1, x_2, \ldots, x_n)$$

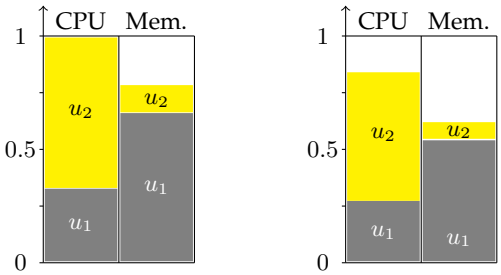
subject to (1)(2).

As mentioned in [7], the number of tasks need to be processed on the server is bounded in the realistic multi-resource environment, i.e.,

$$x_i \le B_i, \quad \text{for} \quad i = 1, \ldots, n. \tag{3}$$

It is well-know that the equality (2) may not be possible if some r_{ij} are zero [3]. It also holds when the number of tasks for every user is finite.

Example 1. Consider a system with of 18CPUs, 36GB RAM, and two users, where user u_1's task requires (1 CPU, 4GB), and user u_2's task requires (3 CPU, 1GB). Assume that $w_{ij} = 1/2$ for all i, j. The DRF mechanism [1] will allocate (6 CPU, 24GB) to user u_1 and (12 CPU, 4GB) to user u_2, giving the allocation in Figure 1(a). With this allocation, each user ends up with the same dominant share $2/3$. Assume the numbers of tasks for users need to be processed are $B_1 = 5$ and $B_2 = 3$, respectively. In DRF with bounded number of tasks, the server will allocate (5 CPU, 20GB) to user u_1 and (9 CPU, 3GB) to user u_2 to maximize system utilization, giving the allocation in Figure 1(b).



(a) Unbounded (b) Bounded

Figure 1: Comparison two cases

For the bounded case, we observe that:

(a) The dominant shares may not be equal. In the above example, the dominant share of user u_1 is $\max\{5 \cdot 1/18, 5 \cdot 4/36\} = 5/9$, while the dominant share of user u_2 is $\max\{3 \cdot 3/18, 3 \cdot 1/36\} = 1/2$.

(b) There may not exist a bottleneck (or saturated) resource for the bounded case, where the bottleneck resource is the resource j satisfying $\sum_{i=1}^{n} r_{ij} x_i = 1$. In the above example, both CPUs and RAM are not saturated.

As mentioned before, Wang, Li and Liang [7] proposed the water-filling algorithm for DRF with bounded number of tasks in heterogeneous cloud computing systems, which can also be used to the single server case here. However, the running time is not polynomial. Note that DRF is a special case of generalized resource fairness (GRF) [12], which includes both DRF and Asset fairness [1]. Therefore, we shall study the generalized resource fairness with bounded number of tasks, which generalizes GRF and DRF with bounded number of tasks, and design an efficient algorithm.

3 Generalized Resource Fairness with Bounded Number of Tasks

As in [12], let $\|\cdot\|_p$ ($p \ge 1$) be a norm on $\mathcal{R}^m$, where

$$\|\mathbf{ws}_i\|_p = (\sum_{j=1}^{m} ws_{ij}^p)^{1/p}, \tag{4}$$

called the normalized share, is the measure of "how much" per task of user i obtains-a scalar that quantifies the weighted-share vector $\mathbf{ws}_i = (ws_{i1}, \ldots, ws_{im})$. For convenience, let

$$NS_i = \|\mathbf{ws}_i\|_p \cdot x_i \tag{5}$$

be the (total) normalized share for user u_i. Therefore, for the multi-resource fair allocation problem with bounded number of tasks, we want to allocate the resources as fair (may not equal) as possible, and the objective is to maximize

system utilization $(x_1, \ldots, x_n)$, as in [1]. The main problem we face is how to define "fair" . The well-known lexicographically max-min fairness [17] is a good choice. Given a feasible allocation, we compute the normalized share for each user. For convenience, let $\mathbf{NS} = (NS_1, \ldots, NS_n)$ be the normalized share vector.

Definition 1. A normalized share vector $\mathbf{NS}$ is called lexicographically max-min (LMM)-optimal, if for every feasible normalized share vector $\mathbf{NS}' = (NS'_1, \ldots, NS'_n)$, $\mathbf{NS}_\tau$ is lexicographically greater than $\mathbf{NS}'_\tau$, where $\mathbf{NS}_\tau$ ($\mathbf{NS}'_\tau$) is the vector obtained by arranging the normalized shares in $\mathbf{NS}$ ($\mathbf{NS}'$) in order of increasing magnitude.

For example, given two feasible normalized share vectors $\mathbf{NS} = (0.8, 0.9, 0.4)$ and $\mathbf{NS}' = (0.6, 0.7, 0.5)$, $\mathbf{NS}$ is not LMM-optimal, as $\mathbf{NS}'_\tau = (0.5, 0.6, 0.7)$ is lexicographically greater than $\mathbf{NS}_\tau = (0.4, 0.8, 0.9)$.

Noting that the system utilization $(x_1, \ldots, x_n)$ increases generally with increasing normalized share vector $(NS_1, \ldots, NS_n)$, we propose the *lexicographically max-min normalized share* (LMMNS) mechanism for the multi-resource fair allocation problem with bounded number of tasks, which is to find the LMM-optimal share vector $\mathbf{NS}$, subject to (1)(3). Clearly, when $p = +\infty$, LMMNS is DRF with bounded number of tasks considered in [7]. When $p = 1$, LMMNS is a generalization of asset fairness [1], where the number of tasks is infinite.

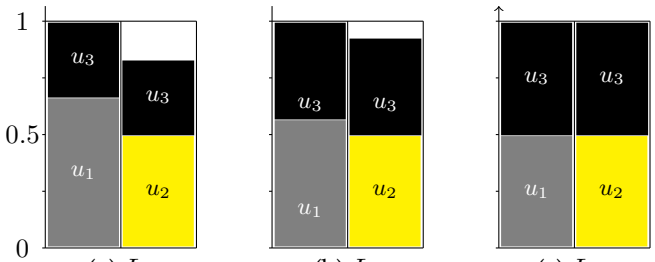
3.1 Examples

Consider a systemwith two resources and three users, which is similar to the example in [12]. The requirements and the number of tasks for three users are given in table 1.

users	u_1	u_2	u_3
Res. 1	0.1	0	0.1
Res. 2	0	0.1	0.1
B_i	10	5	10

Table 1: Requirement matrix

As pointed in [12], it is hard to tell which norm provides an intuitively fairer allocation. However, since the utilitarian and the resource utilization are important criterions in cloud computing systems, we shall compare them of different optimal allocations corresponding to different norms. Figure 2 shows three different optimal allocations corresponding to three different norms L_1, L_2 and L_∞.



(a) L_1 (b) L_2 (c) L_∞

Figure 2: Three allocations under different norms

As seen in Figure 2, although the total number of tasks processed on the server is 15 under three different norms, the resource utilization ratios are different. For the above example, it is reasonable to choose the third allocation under the L_∞ norm, as it has the highest utilization ratio. However, this does not always hold. Consider a systemwith two resources and four users, where the requirements and the number of tasks for four users are given in table 2.

users	u_1	u_2	u_3	u_4
Res. 1	0.45	0.25	0.2	0.1
Res. 2	0.05	0.25	0.3	0.4
B_i	10	10	10	10

Table 2: Requirement matrix

Clearly, both two resources are saturated for the allocation under the L_1 norm, while resource 1 is not saturated for the allocation under the L_∞ norm. The average case of the resource utilization is discussed in Section 8.

3.2 Scheduling algorithm

Although a LMM-optimal solution can be computed approximately by using the water-filling algorithm [7], or by modifying the algorithms in [3], [12], the running time is high. In this subsection, we will use a somewhat different algorithm to find the LMM-optimal solution. Here, if $r_{ij} = 0$ for some i, j, we can allocate the resources in multiple rounds as in [3], [12]. Thus, for purpose of exposition, we will generally restrict our attention to $r_{ij} > 0$ for all i, j throughout this paper. When we violate this convention, we will be explicit about it.

Since the number of tasks for each user u_i is bounded by B_i, in any feasible allocation, the maximum normalized share for user u_i is $NS_i^{max} = \|\mathbf{ws}_i\|_p \cdot B_i$. For a positive number $NS > 0$, let $\mathbf{A}(NS) = (NS_1, \ldots, NS_n)$ be the share vector such that $NS_i = \min\{NS_i^{max}, NS\}$ for each user u_i.

Lemma 1. There exists a positive number NS^* such that $\mathbf{A}(NS^*)$ is the LMM-optimal normalized share vector.

Proof. Let $(NS_1^*, \ldots, NS_n^*)$ be a LMM-optimal normalized share vector and $NS^* = \max_i NS_i^*$. If there is a user u_k satisfying $NS_k^* \neq \min\{NS_k^{max}, NS^*\}$, we have $NS_k^* < NS_k^{\max}$ and $NS_k^* < NS^*$. Consider the user u_h satisfying $NS_h^* = NS^* > NS_k^*$. Let $x_i^* = NS_i^*/\|\mathbf{ws}_i\|_p$, for $i = 1, \ldots, n$. We construct a feasible normalized share vector $\mathbf{NS}' = (NS'_1, \ldots, NS'_n)$, where

$$NS'_i = \begin{cases} \|\mathbf{ws}_h\|_p \cdot (x_h^* - \epsilon_1), & i = h, \\ \|\mathbf{ws}_k\|_p \cdot (x_k^* + \epsilon_2), & i = k, \\ NS_i^*, & i \neq k, h. \end{cases} \quad (6)$$

Here, ϵ_1, ϵ_1 are small enough positive numbers such that $NS'_h > NS'_k$, and $\epsilon_2 \leq \epsilon_1 \min_j r_{hj}/r_{kj}$. Clearly, $\epsilon_2 r_{kj} \leq \epsilon_1 r_{hj}$ for every resource j, which implies that $\mathbf{NS}'$ is a feasible normalized share vector. Thus,

$$NS_k^* < NS'_k < NS'_h < NS_h^*, \text{and } NS'_i = NS_i^*, \text{ for } i \neq k, h.$$

This contradictions the fact $(NS_1^*, \ldots, NS_n^*)$ is a LMM-optimal normalized share vector. ∎

According to Lemma 1, a simple way is to use the binary method to find the maximum NS^* such that $\mathbf{A}(NS^*) = (NS_1^*, \ldots, NS_n^*)$ is a feasible share vector satisfying (1) and (3). However, this naive algorithm is not efficient enough. Our algorithm is described as follows. Using Blum et al.'s method [18] to find the median value NS in $\{NS_1^{\max}, \ldots, NS_n^{\max}\}$, let $NS_i = \min\{NS, NS_i^{\max}\}$ and $x_i = NS_i/\|\mathbf{ws}_i\|_p$ for every user u_i. Clearly, if $\sum_{i=1}^n r_{ij}x_i \leq 1$ for every $j = 1, \ldots, m$, we have $NS^* \geq NS$. Otherwise, we have $NS^* < NS$. We distinguish the following two cases:

Case 1. $NS^* \geq NS$. For each user u_i satisfying $NS_i = NS_i^{\max}$, which implies $NS_i^{\max} \leq NS \leq NS^*$, by Lemma 1, we have $NS_i^* = \min\{NS_i^{\max}, NS^*\} = NS_i^{\max}$. Delete user u_i and decrease the corresponding resources consumed by u_i from the instance.

Case 2. $NS^* < NS$. For each user u_i satisfying $NS_i = NS$, which implies $NS_i^{\max} \geq NS > NS^*$, by Lemma 1, we have $NS_i^* = \min\{NS_i^{\max}, NS^*\} = NS^*$. Thus, these users satisfying $NS_i = NS$ have the same normalized share NS^* in the LMM-optimal solution $(NS_1^*, \ldots, NS_n^*)$, and can be merged into a "dummy" user u_{dum}. Let U_{dum} be the set of users satisfying $NS_i = NS$. Given a possible normalized "share" NS_{dum} of the dummy user u_{dum}, for each user u_i in U_{dum}, it consumes $r_{ij}x_i$-fraction resource of type j, where $x_i = NS_{dum}/\|\mathbf{ws}_i\|_p$. Thus, the dummy user u_{dum} consumes $NS_{dum}\mu_j$-fraction resource of type j, where

$$\mu_j = \sum_{i:u_i \in U_{dum}} \frac{r_{ij}}{\|\mathbf{ws}_i\|_p}.$$

Note that the number of users is reduced by half. Again, use Blum et al.'s method [18] to find the median value NS in $\{NS_i^{\max} | u_i$ is the remaining user and is not the dummy user $\}$. Then, compare NS and NS^* similarly to the above discussion, and reduce the number of users by half correspondingly until that there is only one dummy user. Finally, we will find the user u_τ with maximum value $NS_\tau^{\max}$ such that $NS_\tau^{\max} \leq NS^*$. For each user u_i satisfying $NS_i^{\max} \leq NS_\tau^{\max} \leq NS^*$, set $NS_i^* = NS_i^{\max}$. For each user u_i in U_{dum} such that $NS_i^{\max} > NS_\tau^{\max}$, consider the following linear program (LP):

$$\begin{aligned} &\max \quad NS \\ &\sum_{i:u_i \in U_{dum}} NS \cdot \frac{r_{ij}}{\|\mathbf{ws}_i\|_p} \leq 1 - \sum_{i:NS_i^{\max} \leq NS_\tau^{\max}} r_{ij}B_i, \\ &\text{for } j = 1, \ldots, m. \end{aligned}$$

It is easy to verify that

$$\begin{aligned} NS^* &= \min_j \frac{1 - \sum_{i:NS_i^{\max} \leq NS_\tau^{\max}} r_{ij}B_i}{\sum_{i:u_i \in U_{dum}} r_{ij} \cdot \frac{1}{\|\mathbf{ws}_i\|_p}} \\ &= \min_j \frac{1 - \sum_{i:NS_i^{\max} \leq NS_\tau^{\max}} r_{ij}B_i}{\mu_j} \end{aligned}$$

is the optimal solution to the above LP. For each user $u_i \in U_{dum}$, set $NS_i^* = NS^*$, and then we obtain the LMM-optimal solution $\mathbf{A}(NS^*) = (NS_1^*, \ldots, NS_n^*)$.

Algorithm 1 shows the pseudo-code for LMMNS scheduling algorithm.

Algorithm 1 LMMNS pseudo-code

Step 1. Initialization.
  $\mathbf{rc} = (1, \ldots, 1)$,
  $NS_i^* = 0$, for $i = 1, \ldots, n$,
  $U = \{u_1, \ldots, u_n\}$,
  $\mathcal{NS} = \{NS_1^{max}, \ldots, NS_n^{max}\}$,
  $U_{dum} = \phi$,
  $NS_{dum} = 0$,
  $\mu_j = 0$, $j = 1, \ldots, m$,
Step 2. Using the method in [18] to find the median NS in $\mathcal{NS}$, set $NS_{dum} = NS$ and $NS_i^* = \min\{NS, NS_i^{max}\}$ for $u_i \in U$.
  If $NS_{dum}\mu_j + \sum_{i:u_i \in U} r_{ij}B_i \leq \mathbf{rc}_j, \forall j$, set
    $\mathbf{rc}_j \leftarrow \mathbf{rc}_j - \sum_{\substack{i:u_i \in U, \\ NS_i^* = NS_i^{max}}} r_{ij}B_i$, for $j = 1, \ldots, m$;
    $U \leftarrow U \setminus \{u_i | NS_i^* = NS_i^{max}\}$.
  Else, set
    $\mu_j \leftarrow \mu_j + \sum_{\substack{i:u_i \in U, \\ NS_i^* < NS_i^{max}}} r_{ij} \cdot \frac{1}{\|\mathbf{ws}_i\|_p}$, for every$j$;
    $U_{dum} \leftarrow U_{dum} \cup \{u_i | NS_i^* < NS_i^{max}, u_i \in U\}$;
    $U \leftarrow U \setminus \{u_i | NS_i^* < NS_i^{max}\}$.
Step 3. If $U \neq \phi$, goto **Step 2**;
  else, for each user $u_i \in U_{dum}$, set
    $NS_i^* \leftarrow \min_j \frac{1 - \sum_{i:NS_i^{\max} \leq NS} r_{ij}B_i}{\mu_j}$.
Step 4. Output NS_i^*, for $i = 1, \ldots, n$.

At each iteration k, since we have at most $n/2^{k-1} + 1$ users, deciding whether $NS \geq NS^*$ can be done within $O(mn/2^{k-1})$ time. Thus, the overall running time is $O(m(n + n/2 + n/2^2 + \ldots + 1)) = O(mn)$, which is linear in n when m is a bounded number. When we allow $r_{ij} = 0$, we can execute Algorithm 1 at most m rounds as in [3], [12] to find the LMM-optimal solution. The running time is also linear in n, as m is a bounded number in reality.

4 Allocation Properties

Similarly to [1], [3], the following are important and desirable properties of a fair multi-resource allocation with bounded number of tasks:

1. **Pareto efficiency** (PE). It should not be possible to increase the number of tasks processed on the server without decreasing the allocation of at least another user.

2. **Sharing incentive** (SI). For all users u_i: either $x_i = B_i$ or there exists a resource j such that $r_{ij}x_i \geq w_{ij}$.

3. **Envy-freeness** (EF). For all users u_i: either $x_i = B_i$ or for every user k, there exists a resource j such that

$$\frac{r_{ij}x_i}{w_{ij}} \geq \frac{r_{kj}x_k}{w_{kj}}.$$

4. **Group Strategy-proofness** (GSP). No user can schedule more tasks by forming a coalition with others to misreports their requirements r_{ij} or the number of tasks B_i.

PE is to maximize system utilization subject to satisfying the other properties. SI means that the number of tasks processed for each user u_i is no less than the case where a w_{ij}-fraction of each resource j is allocated to every user. EF requires that for every user u_i such that $x_i < B_i$, she do not envy user u_k when the allocation of u_k is scaled by w_{ij}/w_{kj}. GSP means no user can get a better allocation by lying about r_{ij} or B_i.

Remark. It is straightforward to verify that all the utilization maximization mechanisms including LMMNS satisfy the PE property.

As before, for purpose of exposition, we restrict our attention to $r_{ij} > 0$ for all i, j. Consider the LMM-optimal normalized share vector $\mathbf{NS}^* = (NS_1^*, \ldots, NS_n^*)$ and the corresponding utility vector $(x_1^*, \ldots, x_n^*)$, where $x_i^* = NS_i^*/\|\mathbf{ws}_i\|_p \le B_i$ for $i = 1, \ldots, n$.

Lemma 2. If $x_k^* < B_k$ for user u_k, we have $NS_k^* \ge NS_i^*$ for every $i = 1, \ldots, n$.

Proof. If there is a user u_h satisfying $NS_h^* = \|\mathbf{ws}_h\|_p \cdot x_h^* > \|\mathbf{ws}_k\|_p \cdot x_k^* = NS_k^*$, we construct a feasible normalized share vector $\mathbf{NS}' = (NS_1', \ldots, NS_n')$, as defined in (6), where $NS_h^* > NS_h' > NS_k' > NS_k^*$. It is easy to verify that $\mathbf{NS}_\tau'$ is lexicographically greater than $\mathbf{NS}_\tau^*$, which contradicts the fact that $\mathbf{NS}^*$ is the LMM-optimal normalized share vector. ∎

Theorem 1. LMMNS satisfies the SI property, if and only if $p = \infty$.

Proof. If $x_i^* = B_i$ for every user i, all users are satisfied. If there is a user u_k such that $x_k^* < B_k$, there exists at least one saturated resource. If not, we can allocate $(x_k^* + \epsilon)(r_{k1}, \ldots, r_{km})$ to user u_k without changing the other x_is, where $\epsilon = \min_j(1 - \sum_{i=1}^n r_{ij}x_i^*)/r_{kj}$. Then, we obtain a new normalized share vector $\mathbf{NS}' = (NS_1', \ldots, NS_n')$, where $NS_k' = (x_k^* + \epsilon)\|\mathbf{ws}_k\|_p$ and $NS_i' = NS_i^*$ for $i \ne k$. Clearly, $\mathbf{NS}_\tau'$ is feasible and lexicographically greater than $\mathbf{NS}^*$, which contradicts the fact that $\mathbf{NS}^*$ is the LMM-optimal normalized share vector.

When $p = \infty$, $\|\mathbf{ws}_i\|_p = \max_j r_{ij}/w_{ij}$, for every user i. For convenience, let $j_i = \text{argmax}\, r_{ij}/w_{ij}$, which implies that $r_{ij_i}/w_{ij_i} = \max_j r_{ij}/w_{ij}$. For an arbitrary user u_k such that $x_k^* < B_k$, if there is a resource j satisfying $r_{kj}x_k^* \ge w_{kj}$, then u_k is satisfied. Otherwise, $NS_k^* = r_{ij_k}x_k^*/w_{ij_k} < 1$ and for an arbitrary saturated resource j', we have $r_{kj'}x_k^* < w_{kj'}$. As $\sum_{i=1} w_{ij'} = 1$ and the resource j' is saturated, i.e., $\sum_{i=1}^n r_{ij'}x_{ij'}^* = 1$, there is a user u_h satisfying $r_{hj'}x_h^* > w_{hj'}$, implying that

$$NS_h^* = \|\mathbf{ws}_i\|_p \cdot x_h^* = \frac{r_{ij_h}}{w_{ij_h}} \cdot x_h^* \ge \frac{r_{hj'}}{w_{hj'}} \cdot x_h^* > 1 > NS_k^*.$$

But this contradicts Lemma 2. Thus, for every user u_i, either $x_i^* = B_i$ or there exists a resource j such that $r_{ij}x_i^* \ge w_{ij}$, i.e., LMMNS satisfies the SI property, if $p = \infty$.

For any fixed constant $p < \infty$, it is sufficient to show that LMMNS violates the SI property with an example. Consider a system with of 18CPUs, 18GB RAM, and two users. Assume that $B_1 = B_2 = 18$ and $w_{ij} = 1/2$ for all $i, j = 1, 2$. User u_1's task requires (1, 1), and user u_2's task requires (1, ϵ), where ϵ is a small enough positive number. For convenience, let $\epsilon = 0$ to obtain an approximate result. Clearly, $x_1^* = x_2^*/2^{1/p} = 1/(2^{1/p}+1)$ is a LMMNS allocation. Thus, for $j = 1, 2$, we have

$$r_{1j}x_1^* = x_1^* = \frac{1}{2^{1/p}+1} < \frac{1}{2} = w_{1j},$$

which implies that LMMNS violates the SI property, if $1 \le p < \infty$. ∎

Theorem 2. LMMNS satisfies the EF property.

Proof. For an arbitrary user u_k, if $x_k^* = B_k$, u_k does not envy any user. If $x_k^* < B_k$, by Lemma 2, we have $NS_k^* \ge NS_i^*$ for every user u_i. If user u_k envies another user u_h, u_h must have a strictly higher weighted share of every resource than that of user u_k, i.e., $r_{kj}x_k^*/w_{kj} < r_{hj}x_h^*/w_{hj}$, for $j = 1, \ldots, m$, which implies that

$$\begin{aligned} NS_k^* &= \|\mathbf{ws}_k\|_p \cdot x_k^* = \|(\frac{r_{k1}}{w_{k1}} \cdot x_k^*, \ldots, \frac{r_{km}}{w_{km}} \cdot x_k^*)\|_p \\ &< \|(\frac{r_{h1}}{w_{h1}} \cdot x_h^*, \ldots, \frac{r_{hm}}{w_{hm}} \cdot x_h^*)\|_p = NS_h^*, \end{aligned}$$

where the inequality follows from the monotonicity of $\|\cdot\|_p$. A contradiction. Thus, LMMNS satisfies the EF property. ∎

Theorem 3. LMMNS satisfies the GSP property.

Proof. Denote by $\bar{\mathbf{NS}} = (\bar{NS}_1, \ldots, \bar{NS}_n)$ the LMM-optimal normalized share vector when a coalition of users $\bar{U} \subseteq \{u_1, \ldots, u_n\}$ misreports requirement $\bar{r}_{ik}$ instead of r_{ik} and $\bar{B}_k$ instead of B_k for all $u_k \in \bar{U}$. For an arbitrary user u_k, if $x_k^* = B_k$, user u_k cannot increase its utility by altering the demand vector or B_k. Thus, we assume that $x_k^* < B_k$ for each user $u_k \in \bar{U}$. By Lemma 2, for an arbitrary user $u_k \in \bar{U}$, we have

$$NS_k^* \ge NS_i^*, \text{ for } i = 1, \ldots, n. \tag{7}$$

Let $\bar{\mathbf{NS}} = (\bar{NS}_1, \ldots, \bar{NS}_n)$ be the LMM-optimal solution for the modified system. Denote by $(\bar{x}_1, \ldots, \bar{x}_n)$ the corresponding numbers of tasks processed in the LMMNS solution of the modified system. Thus, for each user $u_k \in \bar{U}$, $\bar{NS}_k = \|\bar{\mathbf{ws}}_k\|_p \cdot \bar{x}_k$, where $\bar{ws}_{kj} = \bar{r}_{kj}/w_{kj}$, and for each user $u_i \notin \bar{U}$, $\bar{NS}_i = \|\mathbf{ws}_i\|_p \cdot \bar{x}_i$, where $ws_{ij} = r_{ij}/w_{ij}$. For an arbitrary user $u_k \in \bar{U}$, since the true utility of user u_k is increased in the modified system, we have

$$r_{ik}x_k^* < \bar{r}_{ik}\bar{x}_k, \text{ for } j = 1, \ldots, m, \text{ and } NS_k^* < \bar{NS}_k. \tag{8}$$

Consider an arbitrary saturated resource j in the original system (there must exist, as in the proof of Theorem 1). As $\sum_{i=1}^n r_{ij}x_i^* = 1$, $\sum_{i:u_i \in \bar{U}} \bar{r}_{ij}\bar{x}_i + \sum_{i:u_i \in U \setminus \bar{U}} r_{ij}\bar{x}_i \le 1$, and $r_{kj}x_k^* < \bar{r}_{kj}\bar{x}_k$ for each user $u_k \in \bar{U}$, there is a user $u_h \in U \setminus \bar{U}$ such that $r_{hj}x_h^* > r_{hj}\bar{x}_h$, which implies that

$$\bar{x}_h < x_h^* \le B_h,$$

and

$$\bar{NS}_h = \|\mathbf{ws}_h\|_p \cdot \bar{x}_h < \|\mathbf{ws}_h\|_p \cdot x_h^* = NS_h^*. \tag{9}$$

Thus, by the optimality of $\bar{\mathbf{NS}} = (\bar{NS}_1, \ldots, \bar{NS}_n)$ and Lemma 2 , we have

$$\bar{NS}_h \ge \bar{NS}_k. \tag{10}$$

Combining (7)-(10), we have

$$\bar{NS}_h \ge \bar{NS}_k > NS_k^* \ge NS_h^* > \bar{NS}_h.$$

A contradiction. Therefore, every user $u_k \in \bar{U}$ cannot increase her utility by altering the requirement vector or B_k, which implies that LMMNS satisfies the GSP property. ∎

5 EFFICIENCY

In the last section, we have proved that LMMNS satisfies some highly desirable fairness properties. Especially, when $p = \infty$, LMMNS satisfies the SI property which is one of the most important fairness properties. For convenience, when $p = \infty$, LMMNS is called *lexicographically max-min dominant share* (LMMDS, for short), following from that $\|\mathbf{ws}_i\|_\infty = w_{ij_i}x_i^* = DS_i$, where j_i is defined as in the proof of Theorem 1.

In this section, we examine the efficiency of LMMNS. In [3], Parkes et al. proved that DRF is good enough as far as social welfare maximization is concerned, where DRF is a special case of LMMDS. In Section 5.1, we present a more accurate result for the LMMDS (or DRF) mechanism. In addition, we analyze the efficiency of LMMDS for the resource utilization in Section 5.2.

5.1 Welfare maximization

As in [3], given an allocation, define its (utilitarian) social welfare as $\sum_i x_i$. The *approximation ratio* of a mechanism is the worst-case ratio between the social welfare of the optimal solution and the social welfare of the mechanism's solution. Note that approximation ratio is equivalent to price of fairness defined in [19]. Parkes, Procaccia, and Shah [3] show that the approximation ratio of DRF is at least m. Noting that the LMMNS mechanism is exactly the DRF mechanism when $p = B_i = \infty$ for $i = 1, \ldots, n$, the approximation ratio of the LMMNS mechanism is at least m, too.

Let $(x_1^*, \ldots, x_n^*)$ be the LMMNS solution throughout this section. Consider a setting with one resource and two users. Assume that $w_{11} = r_{11} = 1/K$, $w_{21} = r_{21} = 1 - 1/K$, and $B_1 = B_2 = K + 1$, where K is a large positive number. The LMMNS mechanism produces a fair allocation with $x_1^* = x_2^* = 1$ and $NS_1^* = NS_2^* = 1$. It is easy to verify that the social welfare of the optimal solution is K obtained by allocating all resources to user u_1. Thus, the approximation ratio is $K/2$, which approaches infinity when $K \to \infty$. Thus, the approximation ratio of LMMNS is infinity. It is easy to verify that the approximation ratio of LMMNS is infinity even if $p = B_i = \infty$ for every i, which implies that the approximation ratio of DRF is infinity, too. The main reason for the bad approximation ratio is that every user has a different weight w_{ij}.

When each user contributes equal amount for every type of resource [1], i.e., $w_{ij} = 1/n$ for every i, j, we obtain some different results. Consider a setting with m resources and n users. Assume that $B_1 = B_2 = K + 1$, the requirement vector of user u_1 is $(1/K, \ldots, 1/K)$ and the requirement vector of user u_i is $(1, 0, \ldots, 0)$ for $i = 2, \ldots, n$, where K is a large positive number. The LMMNS mechanism produces a fair allocation with $x_1^* = K/(1 + (n-1)m^{1/p})$, $x_2^* = \cdots = x_n^* = m^{1/p}/(1 + (n-1)m^{1/p})$, and $NS_i^* = m^{1/p}/(1 + (n-1)m^{1/p})$ for every user u_i. It is easy to verify that the social welfare of the optimal solution is K obtained by allocating all resources to user u_1. Thus, the approximation ratio approaches $1 + (n-1)m^{1/p}$, when $K \to \infty$. When p is small, the approximation ratio is large.

From now on, since when $p < \infty$, LMMNS violates the SI property which is one of the most important fairness properties, we only consider LMMDS which is LMMNS with $p = \infty$.

Theorem 4. When the objective is welfare maximization, the approximation ratio of LMMDS is exactly n, if $w_{ij} = 1/n$ for every i, j.

Proof. Let $(\tilde{x}_1, \ldots, \tilde{x}_n)$ be a social welfare maximized solution. As $(x_1^*, \ldots, x_n^*)$ satisfies the SI property, for each user u_i, either $x_i^* = B_i$, or there is a resource j such that u_i receives at least $1/n$ -fraction of it. Thus, for each $i = 1, \ldots, n$, we have $x_i^* \le n\tilde{x}_i$, implying that $\sum_{i=1}^n x_i^* \le n\sum_{i=1}^n \tilde{x}_i$.

Consider a setting with one resource and n users. For $i = 1, \ldots, n-1$, the requirement of user u_i is $r_{i1} = 1/n$, and the requirement of user u_n is $r_{n1} = 1/n^K$, where K is a large enough positive number. The optimal allocation will give all of resource to user u_n, for a social welfare n^K. In contrast, under LMMDS each user will receive a $1/n$-fraction of the resource, for a social welfare $n^{K-1} + n - 1$. When K grows larger, the approximation ratio of LMMDS approaches

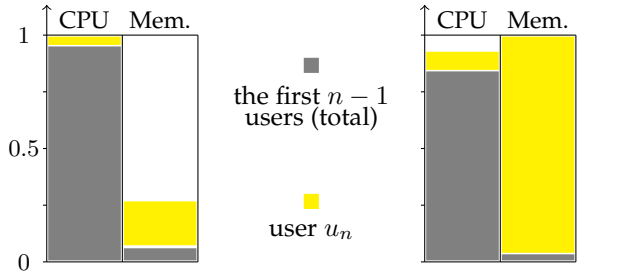
$$\lim_{K\to\infty} \frac{n^K}{n^{K-1} + n - 1} = \lim_{K\to\infty} \frac{n}{1 + \frac{1}{n^{K-2}} - \frac{1}{n^{K-1}}} = n.$$

Thus, for every $\delta > 0$, LMMDS cannot have an approximation ratio better than $n - \delta$ for the social welfare, which implies that the approximation ratio of LMMDS (or DRF) is exactly n. ∎

Since LMMDS satisfies some important highly desirable fairness properties including SI, it is more reasonable to compare LMMDS with the mechanisms satisfying the desirable fairness properties. For example, consider a setting with one resource and n users. Comparing with the mechanisms satisfying SI, the approximation ratio of LMMDS is 1, as any SI mechanism will allocate the resource equally, when $w_{ij} = 1/n$ for every i, j. It is easy to verify that LMMDS is the unique mechanism satisfying PE, SI, EF, and GSP, when $m = 1$. A nature question is that whether LMMDS is the only possible mechanism satisfying PE, SI, EF and GSP, when the number of resources is at least two. Although we do not know other mechanisms, we can compute the approximation ratio of LMMDS, comparing with the mechanisms satisfying certain properties.

Theorem 5. If $w_{ij} = 1/n$, for every i, j, the approximation ratio of LMMDS lies in $[n-1, n]$, comparing with the welfare maximization mechanisms satisfying PE, SI and EF.

Proof. Following from Theorem 4, the approximation ratio of LMMDS is at most n. Let $(\tilde{x}_1, \ldots, \tilde{x}_n)$ be a social welfare maximized solution satisfying PE, SI and EF. Consider a setting with two resources (for example, CPUs and memory) and n users, where $B_i = \infty$ for every user u_i. For $i = 1, \ldots, n-1$, the requirements of user u_i are $r_{i1} = 1/n$ and $r_{i2} = 1/n^K$, and the requirements of user u_n are $r_{n1} = 1/n^{2K}$ and $r_{n2} = 1/n^{K+1}$, where K is a large positive integer. Clearly, $x_i^* = 1/(1 - 1/n + 1/n^K)$ for $i = 1, \ldots, n-1$ and $x_n^* = n^K/(1 - 1/n + 1/n^K)$, for a social welfare $(n^K + n - 1)/(1 - 1/n + 1/n^K)$. In contrast, the social welfare of $(\tilde{x}_1, \ldots, \tilde{x}_n)$ is $n^{K+1} - n^2 + 2n - 1$, where $\tilde{x}_i = 1$ for $i = 1, \ldots, n-1$ and $\tilde{x}_n = n^{K+1} - n^2 + n$. Two allocations are described in Figure 3.



(a) LMMDS allocation (b) Optimal allocation

Figure 3. Comparing two allocations

It is easy to verify that $(\tilde{x}_1, \ldots, \tilde{x}_n)$ satisfies PE, SI and EF. When K grows larger, the approximation ratio of LMMDS approaches

$$\begin{aligned} & \lim_{K\to\infty} \frac{n^{K+1} - n^2 + 2n - 1}{(n^K + n - 1)/(1 - 1/n + 1/n^K)} \\ = \; & (1 - \frac{1}{n}) \lim_{K\to\infty} \frac{n - 1/n^{K-2} + 2/n^{K-1} - 1/n^K}{1 + 1/n^{K-1} - 1/n^K} \\ = \; & n - 1. \end{aligned}$$

Thus, the theorem holds. ∎

Although LMMDS does not perform well on PE, SI and EF in the worst-case scenario, we believe that LMMDS performs well on GSP. Consider a special case when $w_{ij} = 1/n$, for every i, j. Let $(\tilde{x}_1, \ldots, \tilde{x}_n)$ be a solution produced by a social welfare maximized mechanism $\tilde{\mathcal{M}}$ satisfying PE, SI, EF and GSP. If there is a user u_k satisfying $\tilde{x}_k > x_k^*$, there must be a user u_l satisfying $\tilde{x}_l < x_l^*$, following the PE property of LMMDS. Combining the SI property of $(\tilde{x}_1, \ldots, \tilde{x}_n)$, we have

$$r_{kj_k}\tilde{x}_k > r_{kj_k}x_k^* = r_{lj_l}x_l^* > r_{lj_l}\tilde{x}_l \geq \frac{1}{n}.$$

If user u_l claims that its requirement vector is $(r_{l1}/K, \ldots, r_{lm}/K)$, where K is a large positive number approaching infinity. By the optimality of $\tilde{\mathcal{M}}$, the mechanism $\tilde{\mathcal{M}}$ will possibly reallocate some resources of user u_k to user u_l to improve welfare, without violating the PE, SI and EF properties. However, this violates the GSP property of $\tilde{\mathcal{M}}$. Thus, we believe that LMMDS performs well on GSP for welfare maximization. It is challenging to give the tight approximation ratio of LMMDS, comparing with the welfare maximization mechanisms satisfying PE, SI, EF and GSP.

5.2 Resource utilization maximization

In cloud computing systems, the resource managers care more about the resource utilization. For example, Ghodsi et al. [1] shows CPU and memory utilization for the small workload when using DRF compared to Hadoop's fair scheduler (slot). Given an allocation $(x_1, \ldots, x_n)$, let $c_j = \sum_{i=1}^n r_{ij}x_i$ be the utilization rate (or consumption) of resource j. Define *utilization* of $(x_1, \ldots, x_n)$ as $\min_j c_j$, which is the minimum utilization rate of m resources. Similarly to that in [3], define *approximation ratio* of a mechanism as the worst-case ratio between the utilization of the optimal solution and the utilization of the mechanism's solution.

As in the last subsection, we only consider LMMDS, as LMMNS violates the SI property and has a large approximation ratio when $p < \infty$. Clearly, when $m = 1$, LMMDS is an optimal mechanism, because all the resources are allocated, following from the PE property. When $m \geq 2$, LMMDS will not always be optimal. Consider a setting with two resources and two users. Assume that $w_{11} = w_{12} = r_{11} = r_{12} = r_{22} = 1/K$, $w_{21} = w_{22} = r_{21} = 1 - 1/K$, and $B_1 = B_2 = K + 1$, where K is a large positive number. The LMMDS mechanism produces a fair allocation with $x_1^* = x_2^* = 1$, whose utilization is $2/K$. It is easy to verify that the utilization of the optimal solution is 1 obtained by allocating all resources to user u_1. Thus, the approximation ratio is $K/2$, which approaches infinity when $K \to \infty$. It is easy to verify that the approximation ratio of LMMDS is infinity when $B_i = \infty$ for every i, which implies that the approximation ratio of DRF is infinity, too.

When each user contributes equal amount for every type of resource, i.e., $w_{ij} = 1/n$ for every i, j, we obtain some good results.

Theorem 6. When the objective is utilization maximization, the approximation ratio of LMMDS is exactly n, if $w_{ij} = 1/n$ for every i, j.

Proof. Let $(\tilde{x}_1, \ldots, \tilde{x}_n)$ be a utilization maximized solution with utilization $\tilde{c}$. Since LMMDS satisfies the SI property, for each user u_i, either $x_i^* = B_i$, or $r_{ij_i}x_i^* \geq 1/n$. Combining the constraints $\tilde{x}_i \leq B_i$ and $r_{ij_i}\tilde{x}_i \leq \sum_{k=1}^n r_{kj_k}\tilde{x}_k \leq 1$, we have $x_i^* \geq \tilde{x}_i/n$, for $i = 1, \ldots, n$. Let j' be the resource such that $\sum_{i=1}^n r_{ij}\tilde{x}_i$ is minimized, which implies that the utilization of $(\tilde{x}_1, \ldots, \tilde{x}_n)$ is $\tilde{c} = \sum_{i=1}^n r_{ij'}\tilde{x}_i$. Obviously, for $j = 1, \ldots, m$, we have

$$\sum_{i=1}^n r_{ij}x_i^* \geq \frac{1}{n}\sum_{i=1}^n r_{ij}\tilde{x}_i \geq \frac{1}{n}\sum_{i=1}^n r_{ij'}\tilde{x}_i \geq \frac{\tilde{c}}{n}. \tag{11}$$

It implies that the approximation ratio of LMMDS is at most n.

Consider a setting with two resources and n users. The requirements of user u_1 are $r_{11} = 1/n$ and $r_{12} = 1/n$. For $i = 2, \ldots, n$, the requirements of user u_i are $r_{i1} = 1/n$ and $r_{i2} = 1/n^K$, where K is a large positive number. The optimal allocation will give all of resource to user u_1, for a utilization 1. In contrast, under LMMDS, each user will receive a $1/n$-fraction of the resource 1, for a utilization $1/n + (n-1)/n^K$. When K grows larger, the approximation ratio of LMMDS approaches

$$\lim_{K\to\infty} \frac{1}{1/n + (n-1)/n^K} = \lim_{K\to\infty} \frac{n^K}{n^{K-1} + (n-1)} = n.$$

Thus, the approximation ratio of LMMDS is at least n. Therefore, the theorem holds. ∎

As in the last subsection, we compare LMMDS with the mechanisms satisfying certain properties. Similarly, we obtain

Theorem 7. When $w_{ij} = 1/n$, for every i, j, the approximation ratio of LMMDS lies in $[n-1, n]$, comparing with the utilization maximization mechanisms satisfying the PE, SI, and EF properties.

Proof. Following from Theorem 6, the approximation ratio of LMMDS is at most n. Let $(\tilde{x}_1, \ldots, \tilde{x}_n)$ be a utilization maximized solution satisfying PE, SI, and EF. Consider a

setting with three resources and n users. For $i = 1, \ldots, n-1$, the requirements of user u_i are $r_{i1} = 1/n$, $r_{i2} = 1/n^K$, and $r_{i3} = 1/n^{4K}$, and the requirements of user u_n are $r_{n1} = 1/n^{2K}$, $r_{n2} = 1/n^{K+1}$, and $r_{n3} = 1/n^{2K}$, where K is a large positive integer. Clearly, $x_i^* = 1/(1 - 1/n + 1/n^K)$ for $i = 1, \ldots, n-1$ and $x_n^* = n^K/(1 - 1/n + 1/n^K)$, for a utilization $(n-1)/n^{4K}/(1 - 1/n + 1/n^K) + 1/n^K/(1 - 1/n + 1/n^K)$. In contrast, the utilization of $(\tilde{x}_1, \ldots, \tilde{x}_n)$ is $(n-1)/n^{4K} + (n^{K+1} - n^2 + n)/n^{2K}$, where $\tilde{x}_i = 1$ for $i = 1, \ldots, n-1$ and $\tilde{x}_n = n^{K+1} - n^2 + n$. When $K \to \infty$, the approximation ratio approaches

$$\begin{aligned}
&\lim_{K\to\infty} \frac{1}{1 - 1/n + 1/n^K} \frac{n - 1 + (n^{K+1} - n^2 + n)n^{2K}}{n - 1 + n^{3K}} \\
&= (1 - \frac{1}{n}) \lim_{K\to\infty} \frac{n - 1 + n^{3K+1} - n^{2K+2} + n^{2K+1}}{n - 1 + n^{3K}} \\
&= n - 1.
\end{aligned}$$

Thus, the theorem holds. ∎

6 Existing alternative fair mechanisms

6.1 Generalized Competitive equilibrium from equal incomes

In microeconomic theory, as mentioned in [1], a common method for resource allocation is competitive equilibrium from equal incomes (CEEI), where each user receives initially $1/n$ of every resource and subsequently, each user trade her resources with other users in a perfectly competitive market [20]. Formally, CEEI is to maximize $\prod_{i=1}^n x_i$, subject to the capacity constraints. Bonald and Roberts [13] considered the proportional fairness (PF) mechanism, originally from network bandwidth sharing networks [21], which is to maximize $\sum_{i=1}^n \ln x_i$ subject to the capacity constraints. They [13] showed that PF is more preferable to DRF in the dynamic setting. It is easy to verify that PF is equivalent to CEEI.

For the multi-resouce allocation problem with bounded number of tasks, a natural generalization of CEEI (or PF), called generalized CEEI, is to maximize $\prod_{i=1}^n x_i$ subject to the capacity constraints and $x_i \le B_i$, for $i = 1, \ldots, n$. With slight modifications of the method in [1], [13], it is easy to verify that

Theorem 8. Generalized CEEI satisfies the PE, SI and EF properties, and violates the GSP property.

As in the last section, when examine the efficiency of BBF, we have

Theorem 9. When the objective is welfare (or utilization) maximization, the approximation ratio of Generalized CEEI is exactly n, if $w_{ij} = 1/n$ for every i, j.

6.2 Bottleneck based fairness

Dolev et al. [11] proposed the bottleneck based fairness (BBF) mechanism, which is to find a fair allocation $(x_1, \ldots, x_n)$ such that $\sum_{i=1}^n r_{ij} x_i \le 1$ and

for all users $u_i : x_i = 1$, or

there exists a bottleneck resource j^* such that $x_i r_{ij^*} \ge w_{ij^*}$.

Here, we use a natural generalization of the BBF mechanism [11] by replacing e_i with w_{ij^*}. Given an instance of the multi-resource allocation problem with bounded number of tasks, consider a modified instance with $r'_{ij} = r_{ij} B_i$. In the BBF solution $(x'_1, \ldots, x'_n)$ for the modified instance, $x'_i = 1$ implies that $x_i = B_i$ for the original instance. Thus, BBF can produce a BBF solution for the multi-resource allocation with bounded number of tasks.

Notably, Gutman and Nisan [12] described a method to convert a CEEI solution to a BBF solution, which is a generalized CEEI solution. As a CEEI solution can be found in polynomial-time based on convex programming [22], a BBF solution can be found in polynomial-time, too. Obviously, BBF satisfies the PE and SI properties. Although CEEI satisfies the EF property, BBF may violate the EF property, as there may be many BBF solutions when $n > 2$ [11].

Theorem 10. BBF violates the EF property, when $n > 2$.
Proof. It is sufficient to show this with an example. Consider a scenario with with three users and two resources. The requirements of the users are $(1/3, 1/6)$, $(1/2, 1/3)$ and $(1/6, 1/2)$. Clearly, $x_1 = x_2 = x_3 = 1$ is a BBF solution. Since $(1/3, 1/6)x_1 < (1/2, 1/3)x_2$, BBF violates the EF property.

Theorem 11. BBF violates the GSP property.
Proof. It is sufficient to show this with an example appeared in [13]. Consider a scenario with with two users and two resources. The requirements of the users are $(1/2, 1)$ and $(1, 1/2)$. The BBF solution is $x_1 = x_2 = 2/3$. If user u_1 claims $(\bar{r}_{11}, \bar{r}_{12}) = (2/3, 1)$, BBF produces $\bar{x}_1 = 3/4$ and $\bar{x}_2 = 1/2$, yielding $\bar{x}_1(\bar{r}_{11}, \bar{r}_{12}) = (1/2, 3/4) > x_1(1/2, 1)$.

Similarly, we have

Theorem 12. When the objective is welfare (or utilization) maximization, the approximation ratio of BBF is exactly n, if $w_{ij} = 1/n$ for every i, j.

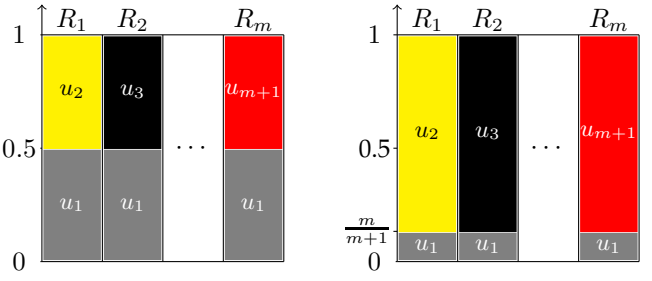
6.3 Comparison with LMMDS

Since BBF violates the EF property and a generalized CEEI solution is a BBF solution, it is sufficient to compare generalized CEEI with LMMDS. As mentioned before, the approximation ratios of two mechanisms are exactly n, when $w_{ij} = 1/n$ for every i, j. It is worth mentioning that LMMDS satisfies the GSP properties, which is violated by generalized CEEI. Moreover, a LMMDS solution can be found within nearly linear time. In contrast, it is hard to find a generalized CEEI solution in polynomial time. Finally, a generalized CEEI solution can unfortunately consist of irrational numbers [22], which implies we can only find an approximation generalized CEEI solution in some cases. In a word, for the multi-resource fair allocation with bounded number of tasks, we argue here that LMMDS is preferable to generalized CEEI.

7 A modified version of LMMNS

In [1], Ghodsi et al. asked that whether DRF is the only possible GSP mechanism satisfying PE, SI and EF. To the best of knowledge, this problem is unsolved, which implies that LMMDS is the only mechanism satisfying all the highly desired fairness properties. However, we do not think that the LMMDS is "fair" enough. Consider a setting with m

resources and $n = m + 1$ users. The requirement of user u_1 is $1/2$ per task for every resource. For $i = 2, \ldots, m + 1$, each task of user u_i require $1/2$ of resource $i - 1$ and zero of others. The bounded number of tasks is 2 for every user. The LMMDS allocation will assign $1/2$-fraction of every resource to user u_1, while the LMMNS allocation with $p = 1$ will assign $1/(m + 1)$-fraction of every resource to user u_1, as shown in Figure 4.



(a) LMMDS (b) LMMNS with $p = 1$

Figure 4: Comparison two cases

From any perspective, in the above example, LMMDS allocate too many resources to user u_1. Obviously, if a user u_i requires the same fraction of every resource, it is more reasonable to allocate $1/n$-fraction of every resource to it, assuming $w_{ij} = 1/n$ for every j. Thus, the LMMNS solution with $p = 1$ (Figure 4(b)) is the fairest allocation for the above example.

Recall that the only drawback of LMMNS with $p < \infty$ is that it violates the SI property. However, this can be overcome by a small modification. Instead, different from LMMNS in Section 3, we will find a LMM-optimal solution $\mathbf{NS}$ which also satisfies $x_i \geq 1/n/r_{ij_i}$ for guaranteeing the SI property. Formally, our modified LMMNS is to find a LMM-optimal solution $\mathbf{NS} = (NS_1, \ldots, NS_n)$ such that

$$\begin{cases} \sum_{i=1}^{n} r_{ij} x_i \leq 1, \text{for } j = 1, \ldots, m; \\ NS_i = \|ws_i\|_p \cdot x_i, \text{for } i = 1, \ldots, n; \\ \dfrac{1}{n r_{ij_i}} \leq x_i \leq B_i, \text{for } i = 1, \ldots, n. \end{cases} \tag{12}$$

Next, we will describe how to find a modified LMMNS solution. For $i = 1, \ldots, n$, let

$$NS_i^{\min} = \|ws_i\|_p \cdot \frac{1}{n r_{ij_i}}$$

be the minimum normalized share for user u_i in any feasible modified LMMNS solution. As in Lemma 1, there exists a positive number NS^* such that $(NS_1^*, \ldots, NS_n^*)$ is the LMM-optimal solution for (12), where

$$NS_i^* = \max\{NS_i^{\min}, \min\{NS_i^{max}, NS^*\}\}.$$

Thus, by a small modification, the progressive filling algorithm [1] can be used to find a modified LMMNS solution. Initially, let $NS_i^* = NS_i^{\min}$. At every step, if there are unused resources, find all the users with smallest normalized share, and increase their normalized share synchronously, until there is a user such that $NS_i^* = NS_i^{max}$, or their normalized share is equal to the second smallest one, or at least one resource is exhausted. Clearly, the running time is $O(n^2)$, assuming m is a fixed constant.

Using the method in Section 4, we can prove

Theorem 13. Modified LMMNS satisfies the PE, SI, EF, and GSP properties.

Theorem 13 implies that DRF is not the only mechanism satisfied these desired propertied, which answers the question proposed in [1]. When examine the efficiency of modified LMMNS, based on the example in Figure 4, we have

Theorem 14. Modified LMMNS is not an optimal welfare maximization mechanisms satisfying PE, SI, EF and GSP.

Proof. We first verify that Modified LMMNS with $p < \infty$ is not an optimal welfare maximization mechanism. Consider a setting with m resources and $n = m+1$ users. The requirement of user u_1 is $1/K$ per task for every resource, where K is a large enough number. For $i = 2, \ldots, m + 1$, each task of user u_i requires 1 of resource $i - 1$ and zero of others. The bounded number of tasks is K for every user. The LMMDS allocation will assign $1/2$-fraction of every resource to user u_1, for a welfare $(K + m)/2$, while modified LMMNS with $p < \infty$ will assign $1/(m^{1/p}+1)$-fraction of every resource to user u_1 for a welfare $K/(m^{1/p} + 1) + m^{1+1/p}/(m^{1/p} + 1) < (K+m)/2$, as K is a large number and p is a fixed constant.

Modified LMMNS with $p = \infty$, i.e., LMMDS is also not an optimal welfare maximization mechanism, which can be verified by the following example. Consider a setting with m resources and $n = m + 1$ users. The requirement of user u_1 is 1 per task for every resource. For $i = 2, \ldots, m + 1$, each task of user u_i requires $1/K$ of resource $i - 1$ and zero of others, where K is a large enough number. The bounded number of tasks is K for every user. The LMMDS allocation will assign $1/2$-fraction of every resource to user u_1, for a welfare $(Km + 1)/2$, while modified LMMNS with $p = 1$ will assign $1/(m + 1)$-fraction of every resource to user u_1 for a welfare $1/(m + 1) + m^2K/(m + 1) > (Km + 1)/2$, as K is a large number and $m^2/(m+1) > m/2$ for any $m \geq 2$. ∎

When the objective is utilization maximization, modified LMMNS with $p < \infty$ is not an optimal mechanism, which can be verified by examining the following example. Consider a setting with 2 resources and $n = 4$ users. The requirements and the number of tasks for four users are given in table 3.

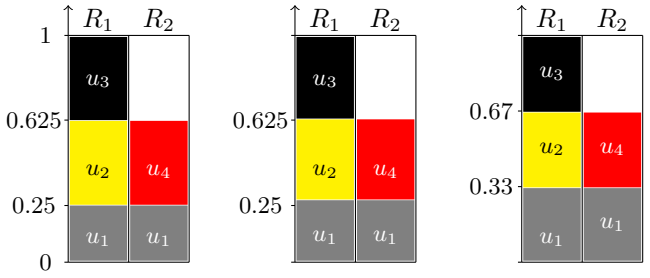
users	u_1	u_2	u_3	u_4
Res. 1	1	1	1	0
Res. 2	1	0	0	1
B_i	1	1	1	1

Table 3: Requirement matrix

LMMDS will assign $1/3$-fraction of every resource to user u_1, for a utilization $2/3$, while modified LMMNS with $p < \infty$ will assign $\max\{1/4, 1/(2^{1/p+1} + 1)\}$-fraction of every resource to user u_1, for a utilization at most

$$\frac{2^{1/p} + 1}{2^{1/p+1} + 1} = \frac{1}{2} + \frac{1/2}{2^{1/p+1} + 1} < \frac{2}{3},$$

where $\log_{3/2} 2 \leq p < \infty$. Figure 5 shows the different allocations under different norms.



(a) $p = 1$ (b) $1 < p < \infty$ (c) LMMDS

Figure 5: Modified LMMNS

Thus, although LMMDS is unfair for some certain cases, we believe that LMMDS is fair and efficient in most cases. Moreover, we guess that the following conjecture holds.

Conjecture. LMMDS is an optimal utilization maximization mechanism satisfying PE, SI, EF and GSP.

8 Experimental Results

For convenience, assume $w_{ij} = 1/n$ for every i, j throughout this section. In every experiment, the requirements r_{ij} of each task are random numbers between 0 and 1, and B_i is a uniform random number in the range $[1/nr_{ij_i}, 1/r_{ij_i}]$ for every user u_i, where $B_i \geq 1/nr_{ij_i}$ implies that user i needs to share the resources. Every data point is obtained by averaging over 50 such simulations. For the welfare (or utilization) maximization, the optimal solution is obtained by solving the linear program, where the software was written in C++ and LINGO 10.

8.1 Welfare maximization

For the results in this subsection, the welfare maximization solution $(\tilde{x}_1, \ldots, \tilde{x}_n)$ is obtained by solving the following linear program:

$$\begin{cases} \max \sum_{i=1}^{n} x_i \\ \sum_{i=1}^{n} r_{ij} x_i \leq 1, \text{for } j = 1, \ldots, m; \\ x_i \leq B_i, \text{for } i = 1, \ldots, n. \end{cases}$$

To quantify the quality of LMMNS for the welfare maximization objective, we compare the LMMNS solution $(x_1^*, \ldots, x_n^*)$ with the optimal solution $(\tilde{x}_1, \ldots, \tilde{x}_n)$, and define the quality as $\sum_{i=1}^{n} x_i^* / \sum_{i=1}^{n} \tilde{x}_i$, which is the inverse of performance ratio defined before and lies in $(0, 1]$. Figure 6-9 show that different qualities with $n = 100$ users by sweeping p for $m = 2, 3, 4, 5$, respectively. Although the quality is $1/100$ in the worst-case scenario as proved in Theorem 4, our experiments show that average quality is at least 0.45. LMMNS performs better for large m, because the probability of requirement vector close to $k(1, \ldots, 1)$ is less. In most (not all) our experiments, the quality is always increasing when p grows, when $m = 2, 3, 4$.

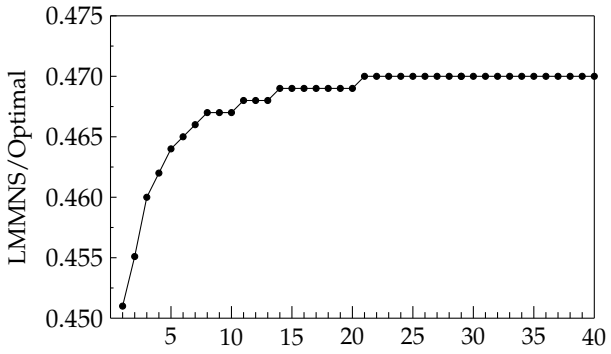


Figure 6: $m = 2$ with varying values p

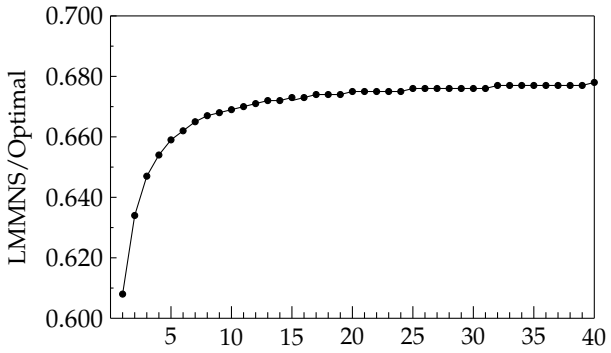


Figure 7: $m = 3$ with varying values p

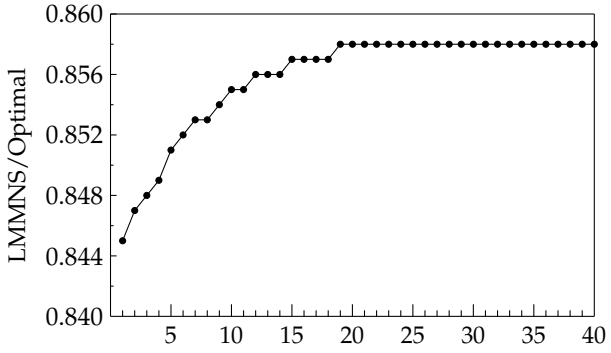


Figure 8: $m = 4$ with varying values p

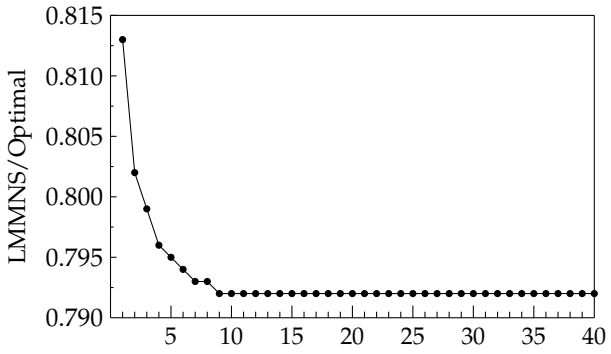


Figure 9: $m = 5$ with varying values p

To verify Theorem 5, we compare the LMMDS solution $(x_1^*, \ldots, x_n^*)$ with the optimal solution $(\tilde{x}_1, \ldots, \tilde{x}_n)$ satisfying the SI property, which is obtained by solving the

following linear program:

$$\begin{cases} \max \sum_{i=1}^{n} x_i \\ \sum_{i=1}^{n} r_{ij} x_i \leq 1, \text{for } j = 1, \ldots, m. \\ r_{ij_i} x_i \geq 1/n, \text{for } i = 1, \ldots, n; \\ x_i \leq B_i, \text{for } i = 1, \ldots, n. \end{cases}$$

Similarly, the quality of LMMDS for welfare maximization is defined as $\sum_{i=1}^{n} x_i^* / \sum_{i=1}^{n} \tilde{x}_i \in (0, 1]$. Figure 10 shows that different qualities with n varying from 100 to 800 for $m = 2, 3, 4, 5$, respectively. Although the quality is $1/(n-1)$ in the worst-case scenario as proved in Theorem 5, our experiments show that average quality is at least 0.9. The quality of LMMDS is increasing when p grows for $m = 2$.

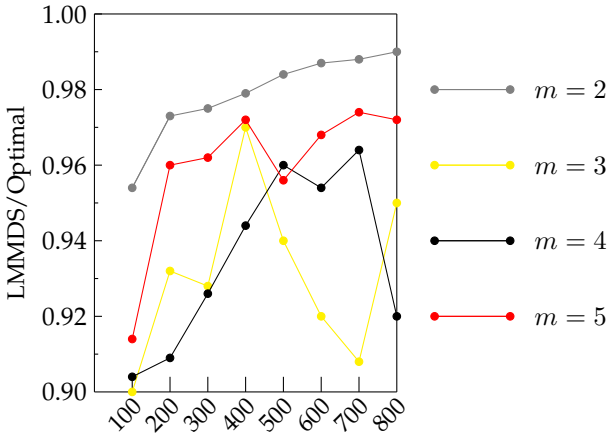


Figure 10: The qualities of LMMDS with varying n

8.2 Utilization maximization

Similarly, the utilization maximization solution can be obtained by solving the following linear program:

$$\begin{cases} \max c \\ c \leq \sum_{i=1}^{n} r_{ij} x_i \leq 1, \text{for } j = 1, \ldots, m; \\ x_i \leq B_i, \text{for } i = 1, \ldots, n. \end{cases}$$

To quantify the quality of LMMNS for the utilization maximization objective, we compare the objective value c^* of the LMMNS solution with the objective value $\tilde{c}$ of the optimal solution, and define the quality as $c^*/\tilde{c}$, which is the inverse of performance ratio defined before and lies in $(0, 1]$. Figure (11-14) show that different qualities with $n = 100$ users by sweeping p for $m = 2, 3, 4, 5$, respectively. Although the quality is not good enough in the worst-case scenario, as in Theorem 6, our experiments show that average quality is at least 0.84.

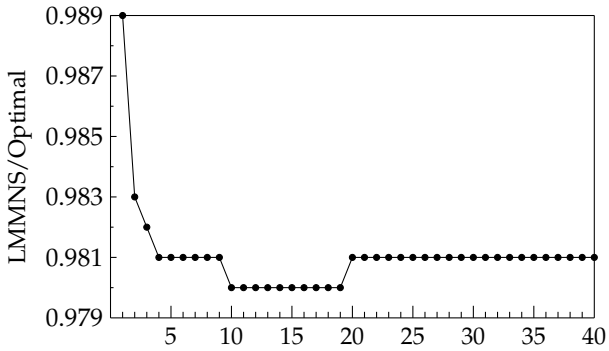


Figure 11: $m = 2$ with varying values p

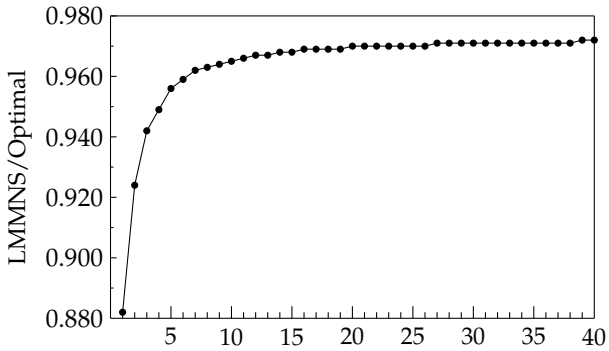


Figure 12: $m = 3$ with varying values p

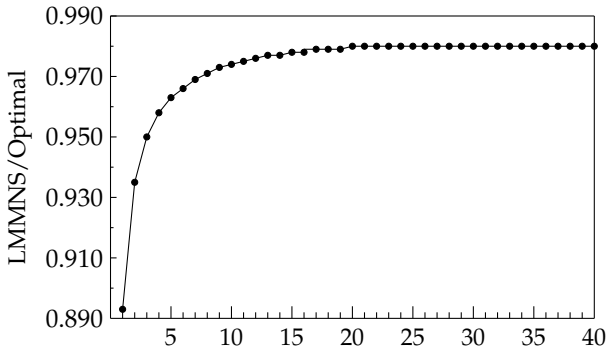


Figure 13: $m = 4$ with varying values p

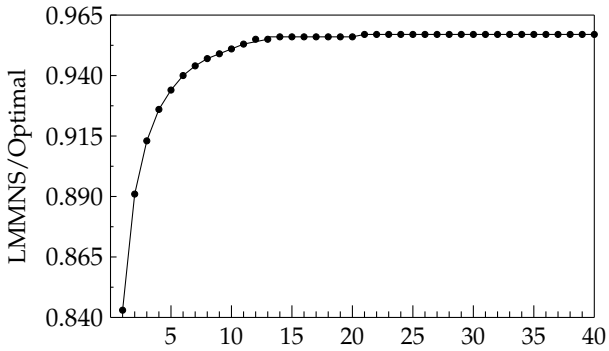


Figure 14: $m = 5$ with varying values p

As in the last subsection, we compare LMMDS with the utilization maximization solution satisfying the SI property, which can be obtained by solving the following linear

program:

$$\begin{cases} \max c \\ c \leq \sum_{i=1}^{n} r_{ij} x_i \leq 1, \text{for } j = 1, \ldots, m. \\ r_{ij_i} x_i \geq 1/n, \text{for } i = 1, \ldots, n; \\ x_i \leq B_i, \text{for } i = 1, \ldots, n. \end{cases}$$

Similarly, the quality of LMMDS for utilization maximization is defined as before. Figure 15 shows that different qualities with n varying from 100 to 800 for $m = 2, 3, 4, 5$, respectively. Although the quality is $1/(n-1)$ in the worst-case scenario as proved in Theorem 7, our experiments show that average quality is at least 0.90. LMMDS performs better when $m = 2$.

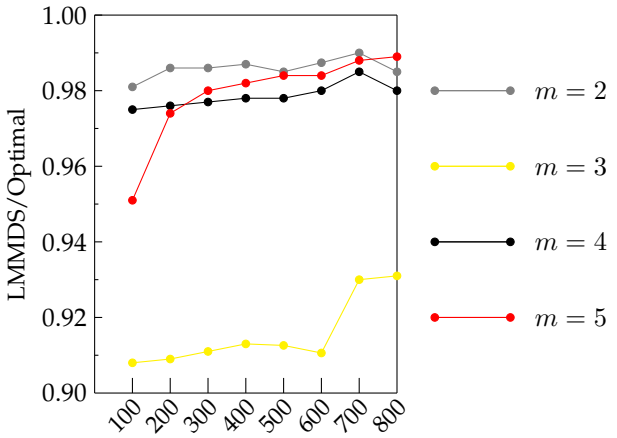


Figure 4. The qualities of LMMDS with varying n

9 Discussions and Future Work

We have presented a linear-time algorithm for the LMMNS mechanism, which a natural generalization of the well-known DRF mechanism. One important direction is to generalize our algorithm to multiple heterogeneous servers [7]. In addition, we have presented the approximation ratios of LMMDS, where the approximation ratio is somewhat different from that defined in [3]. It is still open whether LMMDS (or DRF) is an optimal utilization maximization mechanism satisfying PE, SI, EF and GSP.

As mentioned in [13], proportional fairness (PF) is preferable to DRF under the assumption that the population of jobs in progress is a stochastic process. It is interesting to design an efficient algorithm to find a PF (approximate) solution for multi-resource allocation with bounded number of tasks, which can also be used to the dynamic environment.

Acknowledgment

The work is supported in part by the National Natural Science Foundation of China [Nos. 11301466, 61170222], and the Natural Science Foundation of Yunnan Province of China [No. 2014FB114].

References

[1] A. Ghodsi, M. Zaharia, B. Hindman, A. Konwinski, S. Shenker, and I. Stoica, Dominant resource fairness: fair allocation of multiple resource types. In Proceedings of the 8th USENIX Conference on Networked Systems Design and Implementation, NSDI'11, pp. 24-24, 2011.
[2] C. Joe-Wong, S. Sen, T. Lan, and M. Chiang, Multi-resource allocation: Fairness-efficiency tradeoffs in a unifying framework. In Proceedings of IEEE INFOCOM, pp. 1206-1214, 2012.
[3] D.C. Parkes, A.D. Procaccia, and N. Shah, Beyond dominant resource fairness: extensions, limitations, and indivisibilities. In ACM Conference on Electronic Commerce, pp. 808-825, 2012.
[4] A.A. Bhattacharya, D. Culler, E. Friedman, A. Ghodsi, S. Shenker, and I. Stoica, Hierarchical scheduling for diverse datacenter workloads. In Proceedings of the 4th Annual Symposium on Cloud Computing, SOCC'13, Article No. 4, 2013.
[5] Y. Zeldes and D. G. Feitelson, On-line fair allocations based on bottlenecks and global priorities. In Proceedings of the 4th ACM/SPEC International Conference on Performance Engineering, ICPE 13, pp. 229-240, 2013.
[6] I. Kash, A. Procaccia, and N. Shah, No agent left behind: dynamic fair division of multiple resources, Journal of Artificial Intelligence Research, 51, pp. 351-358, 2014.
[7] W. Wang, B. Li, and B. Liang, Dominant resource fairness in cloud computing systems with heterogeneous servers, In Proceedings of IEEE INFOCOM, 2014.
[8] C.-A. Psomas and J. Schwartz, Beyond beyond dominant resource fairness: indivisible resource allocation in clusters, Tech Report Berkeley, 2013.
[9] E. Friedman, A. Ghodsi, C-A. Psomas, Strategyproof allocation of discrete jobs on multiple machines, in Proceedings of the fifteenth ACM conference on Economics and computation, pp. 529-546, 2014.
[10] A.D. Procaccia, Cake cutting: not just childs play, Communications of the ACM, 2013.
[11] D. Dolev, D. G. Feitelson, J. Y. Halpern, R. Kupferman, and N. Linial, No justified complaints: on fair sharing of multiple resources. In Proceedings of the 3rd Innovations in Theoretical Computer Science Conference, ITCS'12, pp. 68-75, 2012.
[12] A. Gutman and N. Nisan, Fair allocation without trade. In Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems, AAMAS'12, pp. 719-728, 2012.
[13] T. Bonald, J. Roberts, Enhanced cluster computing performance through proportional fairness, Performance Evaluation 79, 134-145, 2014.
[14] T. Bonald, J. Roberts, Multi-resource fairness: Objectives, algorithms and performance, arXiv:1410.0782, 2014.
[15] S. M. Zahedi, and B. C. Lee, REF: Resource elasticity fairness with sharing incentives for multiprocessors, In Proceedings of the 19th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS), pp. 145-160, 2014.
[16] STang, Z. Niu, B. Lee, and B. He, Multi-resource fair allocation in pay-as-you-go cloud computing, manuscript, 2014.
[17] N. Megiddo, Optimal flows in networks with multiple sources and sinks, Mathematical Programming 7(3), pp. 97-107, 1974.
[18] M. Blum, R.W. Floyd, V. Pratt, R.R. Rivest, R.E. Tarjan, Time bounds for selection, Journal of Computer and System Sciences 7(4), 448-461, 1973.
[19] D. Bertsimas, V. F. Farias, and N. Trichakis, The price of fairness, Operations Research 59(1), pp. 17-31, 2011.
[20] H. Moulin, Fair division and collective welfare, The MIT Press, 2004.
[21] F. P. Kelly, A. K. Maulloo, and D. K. H. Tan, Rate control for communication networks: Shadow prices, proportional fairness and stability, The Journal of the Operational Research Society, 49(3), pp. 237-252, 1998.
[22] B. Codenotti and K. R. Varadarajan, Efficient computation of equilibrium prices for markets with leontief utilities. In ICALP, pp. 371-382, 2004.